Multilayered paper prototyping for user concept modeling

Emma Tonkin
UKOLN, University of
Bath, UK
e.tonkin@ukoln.ac.uk

Abstract

This paper describes an investigation of user-centred design methodologies intended to apply to metadata or information architecture evaluation and deployment. The primary focus of this work is investigation of user conceptual models and comparison with formally architected models. We describe related work, primarily from the domain of information architecture, such as free-listing, contextual enquiry, card-sorting and evaluation, and then describes the design, initial evaluation and practical use of a multi-stage prototyping method designed for elicitation of user knowledge and concepts of a domain, common conceptual models in that domain and the objects, collections and relations between objects considered relevant by users. A simple approach to the analysis of results is described.

Keywords: metadata; usability evaluation; paper prototyping

1. Introduction

Recent work in the area of Dublin Core application profile design and development has given rise to a great deal of spirited discussion, led in part by a shift towards the creation and publication of data models that explicitly cover multiple entities and relations. Developers, users and architects need to communicate effectively about real or perceived advantages and disadvantages of such data models, but this has proven to be difficult, leading to some concern of a rift between The following work was sparked by the apparent need to discover fast, hands-on prototyping mechanisms that may be hoped to facilitate exploration of the problem space. These may also be useful for teaching and learning about the relevant concepts, but the primary aim is to find a technology-agnostic and flexible means of expressing user conceptions of structure and concepts such as entities, relationships or hierarchy. Prototyping mechanisms already exist that may be applicable for flat data structures such as taxonomies or simple flat records. However, the explicit handling of concept models within application profiles requires evaluators to develop methods, or adapt existing approaches, in order to facilitate exploration and evaluation of these standards with appropriate user groups.

In general, information architecture has no generally accepted methodology for user-centred design. Sinha and Boutelle (2004) suggest that 'contextual enquiry, ethnographic methods, and card sorting' are used, although it 'remains difficult to go from user research to the design itself'. Several approaches will be briefly summarised in this paper. Conceiving information architecture as a user-centered process is complicated by the fact that designers must balance a number of issues, enumerated by Sinha and Boutelle as: the need to develop an understanding of user conceptual structures; the need to incorporate understanding of business goals and concerns, and the need to insure that the design is neither quickly rendered obsolete, nor designed in too inflexible a manner to incorporate future additions of content and functionality.

1.1. User Conceptual Models

A simple conceptual model describes user perceptions of how an object or system operates. In general, conceptual models are described as difficult for users to comprehend, because they represent abstract generalisations. In discussion of the relation between scenarios, task models and conceptual models, Sutcliffe (2003) quotes Rosch et al (1976) in stating that people 'form

categorical abstractions naturally', in comparison to Hampton's (1988) statement that people 'are less efficient in forming categories of concepts and functions'. As a result, Sutcliffe argues that users may find difficulty in reasoning about conceptual models, even very simple examples such as data flow diagrams. Furthermore, an abstraction learnt in the absence of examples and scenarios may present specific difficulties in terms of evaluation - without provision of such context of use, the tendency is to generalise on the basis of an abstract model. Sutcliffe provides the example of validation of a sample abstract class: birds. The statement 'All birds can fly' is likely to be accepted as valid by many, until a relevant counterexample is provided, bringing into play more specific knowledge about the area. For example, one might ask, 'Is a penguin a bird?'

From the perspective of the investigator, an understanding that reasoning directly about conceptual models is difficult in the abstract implies that exploration of a conceptual model is one that is greatly facilitated by the existence of examples and scenarios of use.

1.2. One Model To Bind Them?

The perception that a single conceptual model is shared by the designers, the developers and the users is very likely to be inaccurate. Many seemingly simple interfaces (such as the ingest interface for Flickr.com) conceal a complex data model, and this is simply an artifact of good design practices.

A 'power user' of Flickr is able to make use of the site's full functionality, creating complex and multilayered relations between images, collections of images, even providing a mechanism by which images may be searched by geographic locality. A novice user can treat the site as nothing more than a digital photograph album, 'pasting' images one by one into the album, and labeling them by typing a title or description on the 'sticky patches' above and below each image. These two prototypical users may well have a very different viewpoint on what Flickr is and what it does; were one to interview Flickr users en masse, would two different conceptual models emerge, one an extended version of the simple model described by the other?

The key concept here is not one of ubiquity, but of compatibility. That the user's perception of the model in which he or she is working is not that of the designer is not automatically problematic, as long as this does not lead the user into misunderstanding the interface or the system functionality, and as long as any information that the user provides to the system is not compromised by this simplified understanding. If simplified forms of the internal model suffice in practical terms, then identification and enumeration of these simplified models may be a useful step in identifying essential information and interface points, as well as easy points of entry for novice users of the system.

1.3. Data, Logic, Physical Structure and the User

It is useful to consider the conceptual model and application profile by analogy to another domain, chosen as an example; the world of enterprise data modeling. We will speak in EDM terms for convenience, although this should be taken to suggest no preference in terms of methodology.

In this domain, a plurality of data models are generally produced for any given problem. Although the specific names and functions of these data models vary, one may think in terms of a conceptual model, a logical data model and a physical data model (ANSI, 1975). A conceptual schema describes the semantics of a domain. It could be considered to be well-adapted for direct user testing, particularly task-based user testing methods, since the conceptual model permits the viewer to explore the types of queries that can be successfully handled by the system; however, conceptual schema understanding is very dependent both on application domain knowledge, and on information systems domain knowledge – such as an understanding of the modeling formalism chosen (Khatri et al, 2006). In the specific instance of certain present application profiles, the handling of elements that are opaque to the user except in effect – such as, under some circumstances, identifiers or relations – may muddy the waters further. In EDM, the

specific functions performed by these technical mechanisms would be discussed within a logical data model, or within a physical data model. Whilst independent of specific technical implementation, a logical data model includes information put in place to enable a detailed view of the sets of data involved (entities, relationships, and attributes). A physical data model includes database-specific information such as, in the case of a relational database, tables, indexes and keys or, in the case of a Semantic Web application, identifiers, relationships, etc.

One might see this distinction as a useful one to keep in mind during the evaluation of a data model. Where an explicit distinction between these areas of functionality is not made, or an explicit series of directives on the area of functionality between 'administrative' and conceptual metadata is not given - in short, where infrastructure and user-facing functionality are presented on the same level - the problem of understanding the model is further complicated. As a result, the burden of on the reader in both the implementation and application domain is increased, as the reader must make these distinctions him- or herself as part of interpreting the model provided.

1.4. Critiques of Prototyping in the Conceptual Domain

A criticism often leveled at the direct use of usability evaluation of (or data collection surrounding) conceptual structures is that in general, what is achieved is primarily evaluation of the interface itself. There is a level of justice to this; the conceptual model represented within an interface design may not (indeed, often should not) closely resemble the logical data model, and hence criticisms leveled at the interface may not be easy to apply to the logical model. They may apply directly to the conceptual model, however. In usability testing of software engineering approaches such as model-view-controller, methodologies often allow for the possibility of refinement of all elements of the model (Sousa et al, 2005). In general, many classes of usability flaw may be traced to the model; for example, Neilsen's heuristic (Neilsen and Molich, 1990) regarding the match between systems and the real world recommends that "the system[...] speak the users' language, with words, phrases and concepts familiar to the user"; this may well point to inconsistencies between the concepts mapped in the data model and those familiar to the user.

A second valid criticism of task-based evaluation making use of prototype software is that the interface chosen, if it is poor, will be detrimental to achieving a good evaluation. That said, the fact that poor interfaces may reduce the effectiveness of user evaluation methodologies is well-understood in human-computer interaction literature, and resulted in a greater focus on paper prototyping methods such as the use of simple drawings or representations on a whiteboard, a set of cards or piece of paper. Snyder (2003) notes that paper prototypes are "less intimidating", achieve a "more creative response" and discourage "nitpicky feedback, because it's obvious that you haven't specified the look yet."

The question of how issues identified via user testing may be traced back to an element of a design, to the conceptual model, to the interface layout or its functional design, is nonetheless a significant point. Diagnosis of where a given issue originated and how it may be solved is non-trivial - and indeed a given complexity may result from the interaction of several layers within the design. Finding a solution may require detailed analysis and exploration of the problem space, perhaps through further prototyping, to identify effective solutions.

2. Review of User-Centred Design Methods Applied in Information Architecture

Several approaches taken toward enabling user-centred design in the domain of information architecture have been mentioned earlier in this paper; these are briefly summarised here.

2.1. Ethnographic Methods and Contextual Enquiry

Ethnography, a term originally used to refer to a branch of anthropology devoted to the study of human society, is used in the human-computer interaction world to refer to the general area of researching human activity through study of user activity in context, observation, interview

techniques and examination of related artifacts. It is a large genre of related research methods, with the general advantage of being extremely powerful, containing little bias and allowing for the unexpected. In general, the focus cast by ethnographic study on the user's environment and context leads to findings that are both useful and unexpected. The downside of the approach is the fact that it is time-consuming, often more expensive than less detailed 'discount' approaches, often requires many participants and the results that are produced are sometimes hard to validate or interpret, and are often difficult to apply directly to a design process.

Contextual enquiry is one of many specific methodologies offering a formal framework enabling the designer to learn about the users who will be making use of the software, and the environment in which they work or live (Beyer and Holtzblatt, 1998). It incorporates a number of ethnographic methods, adapting the approach to be essentially oriented around asking questions, interviewing the user (Rose et al. 1995).

2.2. Free-Listing Exercises

The free-listing technique is borrowed from the domain of cognitive anthropology (Sinha and Boutelle, 2004); it is an approach designed to enable investigators to gain an overview of the scope and boundaries of a content domain, through investigation of user perceptions of the domain. It can also be applied to investigate the level of consistency between participant responses and the level of domain familiarity of a given participant (Sinha, 2003).

The method may be applied within an interview or as a written exercise; the participant is simply asked to "name all the X's you know"; for example, an investigator interested in producing an image-search facility might start by asking participants to name all the types of image that they can think of; an investigator looking at library searching might query participants about book genres. Not only will this provide the most common vocabulary terms applied within that group, which may be useful for later stages of work, but it will also provide an estimate of the types of image (or genre) that are most commonly understood and are most psychologically salient (eg. the most commonly listed, and the terms listed earliest).

Consistency of response will alter depending on domain scope and nature, and on the domain knowledge of participants; approximately 30 participants are usually recommended (Sinha, 2003).

2.3. Card Sorting

Amongst other uses of the technique, card sorting is probably the most common approach to eliciting information about site navigation, taxonomy design and menu structure. It is cheap, requiring only a simple stack of file cards, is quite simple to apply, and represents an easy and effective means of eliciting opinions from individuals or groups of users on the subject of the groupings of terms that they feel 'make sense' – and may as a consequence be expected to be more intuitive in practice. Card sorting is related to the practice of building affinity diagrams (Kawakita, 1991).

Two primary methods for performing card sorts are defined by Spencer and Warfel (2009). In open card sorting, participants are provided with a set of cards that are pre-labeled with information (such as terms or site content). They are then asked to sort them into the groups that they find most appropriate, and then to label each group. In closed card sorting, by contrast, participants are provided with a pre-established set of label terms defining the 'primary groups'; they are then asked to sort the labeled cards into this existing structure. This latter method is primarily of use where information must be added to pre-existing structures.

Card sorting brings with it a number of limitations; Spencer and Warfel (2009) note that it does not support task-based analysis, and the groupings that are provided by users may break down when viewed within the context of a specific task. However, in principle a card-sorting approach may be used in tandem with task-based evaluation – see for example Spencer (2003). Secondly, complex or poorly-understood content will be difficult for participants to work with; some level

of knowledge of the application domain is likely to be required. Thirdly, plain card sorting supports a limited set of relations; it permits participants to indicate the membership of a card set to a given group, but the results are generally hierarchical in nature and hence it is not possible to, for example, describe a given card as linked to multiple groups or categories. This issue may be mitigated by various means; for example, it is possible to build up complex relations through cross-linking cards – for example, by means of colored dots placed on cards to indicate links. This, however, increases the memory load on the user and may be difficult to work with beyond very simple examples.

3. Prototyping Methods for Complex Data Structures

In order to investigate user preconceptions of information within a domain, and to provide interface surrogates for the purpose of investigating possible conceptual models, we first investigated the use of a variant on card sorting (initially using cards and then using sticky notes to simplify transportation of the resulting designs). However, several practical issues quickly became apparent. Firstly, multi-level models (for example, FRBR's Manifestations, contained within Expressions and encapsulated within a Scholarly Work) could not easily be represented in this manner. Secondly, investigation of relationships between items or elements was not easy to visualise and to revise. Thirdly, the cards themselves – or the surface on which they were grouped – rapidly gained many annotations. Initial experience with this method also showed that participants modeling complex data structures were forced to 'overload' cards with multiple meanings - for example, a given card could represent a term/metadata element, a relation between terms, or an entity. This suggested that the resources provided were insufficient to enable fluent expression of the different characteristics of a given model.

FIG. 1. A blank workspace for multilayered paper prototyping

We therefore produced a simple variant on the principle of card sorting. Instead of using cards on a flat surface, a large paper sheet was provided along with sticky notes; a variety of shapes and colors are provided, simplifying participants' task of developing an encoding scheme, or expressing his or her preferred encoding schemes. Due to the space limitations and the 'busy' nature of results achieved using this prototype, we refined it further to include the notion of a multi-layered work surface; several large transparent plastic (acetate) sheets are layered over a large sheet of thin card acting as a base. Sheets are simply clipped onto the card, so that additional sheets can be inserted or sheets reorganised if necessary. Non-permanent overhead projector pens are provided to allow participant to annotate the sheets directly; a whiteboard eraser enables these annotations to be removed or amended.

This approach was chosen, rather than simply working with a standard such as UML, for several reasons. Firstly, UML itself is a difficult concept for most stakeholders. Secondly, the more technical or complete an artifact appears, the greater the disinclination to make changes or volunteer opinion – much as with evaluation of a fully-implemented interface, participants are often disinclined to admit disagreement, state their opinions directly or propose alternatives. Paper prototyping methods are often most productive when the stakes are seen to be low – the participant is invited to explore the possibilities and reassured that within the scope of the session there are no 'right' or 'wrong' answers – and when the tasks that they are given are easily understandable, visual and appear to require no specialist domain knowledge.

3.1. Method

This method may be used either with a single participant, or within a group. Group dynamics have the disadvantage that they may tend to produce results that are not representative of participant consensus, especially when one participant is a domain expert; however, at times, such as during workshops, the group approach may be preferable. With large numbers of participants the group approach may be the only means of ensuring that investigators are able to devote sufficient time to observation and interaction with the participant(s), an important aspect of this approach.

The first key steps of paper prototyping in general are identification of appropriate stakeholders or user groups and the tasks of most relevance to the system and the users. In general, these steps are considered prerequisite to the development of application profiles or data models, so this information is generally already available. That said, it is likely that as understanding of the system and its environment progresses, these key details will also be amended.

After a brief introduction to the concept of user testing, each participant should be provided with a brief introduction to the testing method - one approach to introducing the method is discussion of a simple example from an unrelated domain. An alternative is simply to introduce each stage as a separate task for the user to complete. A series of sample objects are described to the participant; these should be representative examples of the type of information for which a conceptual model is required, including some difficult or complex examples that may be appropriate to encourage the user to challenge their preconceptions about the domain. These examples are provided (or copied) onto small sticky notes and placed on the lowest transparent sheet, the others folded away initially to allow the participant to work.

Initially, the participant is asked whether it is possible to sort these objects into groups; following this, the types of relationship that causes the objects to be grouped are discussed. With complex types of object there will often be several; for example, within a group of objects a participant may state that one object is a version of another, whilst a third is an adaption, and a fourth is thematically similar. The participant is then asked to draw derived groups (entities) and suggest some descriptive terms to explain the ways in which the objects or groups are linked. As sheets become crowded, the participant is invited to add another to the stack; the ability to reorganise sheets is sometimes useful.

A third stage of exploration of the model created by the participant, once he or she is familiar with the types of objects under discussion, is to provide, suggest or ask the participant to suggest some tasks that a user might want to achieve; for example, browsing through objects to find all examples related to a given theme. This allows the participant to subjectively evaluate the decisions that he or she has made, as well as providing input on users' expectations of relevant tasks and approaches to accessing the information. The investigator may wish to take a turn solving a task proposed by the participant, in order to test their understanding of the model.

Overall, the exploration shared between the investigator and participants combines elements of card sorting and free-listing, in order to describe a structure that can then be explored through task-based evaluation.

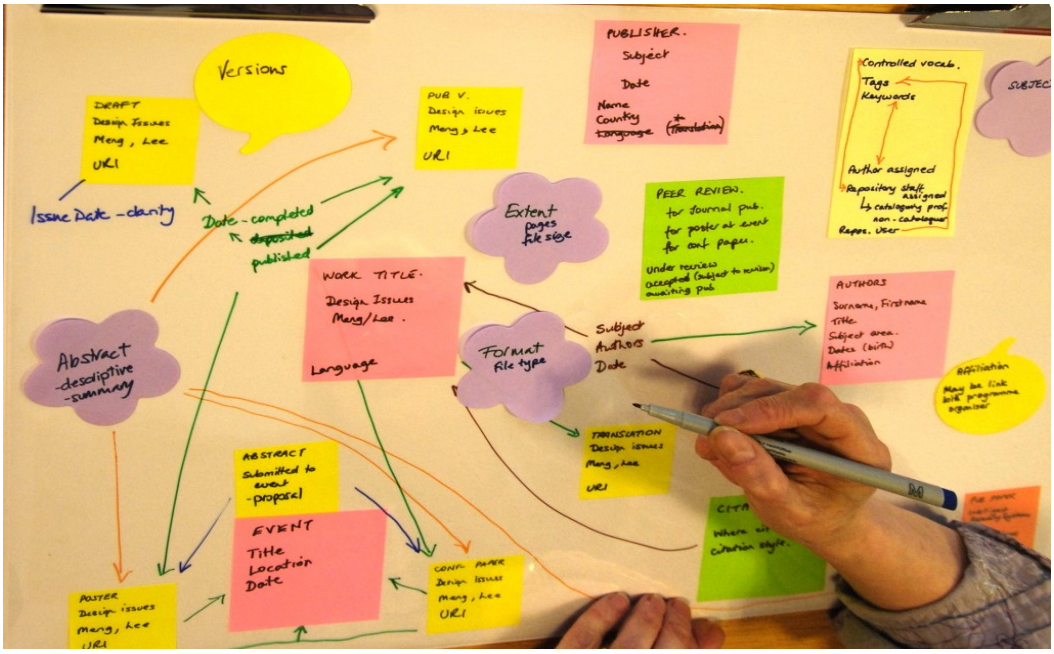

FIG. 2. A sample session encompassing resources, links, entities and agencies

3.2. Analysis of results

As with card sorting, the analysis of the results is potentially a time-consuming problem, mitigated by setting constraints on the task provided to the user, and ensuring that it is clearly and simply described. Sorting of objects can be treated as a standard card-sorting problem, with the caveat that the expected grouping may have been considered, but diagrammed in a different manner. A detailed discussion describing the analysis of card-sorting activities is available from Spencer (2003).

Relationships, entities and properties of groups, all of which may be discussed within the context of a given session, may be treated similarly. Terms applied to each may be elicited and listed alongside a frequency count of appearances, to develop an overall estimation of most preferred terminology. Any type of element from the overall model may be investigated in this way; if elements are functionally equivalent but named differently, then it may be understood as two appearances of a single semantic, with two possible terms associated to it.

Exploration of the model, once created, may be treated as a slightly simplified form of task analysis, and evaluated accordingly. This step is primarily one of establishing the limitations of the model as described, in effect simply by asking questions and seeing whether they may be answered using the structure as laid out.

A great deal of the value of this approach is in close communication with the participants, and the ability to interview users as they develop and explore a concept model. That said, it is important that the analysis and results are carried out in an objective and clear manner; unclear reporting may give rise to contentious debate. As Snyder (2003) puts it, "It's natural for us to filter information through our own set of ideas and prejudices, but this subjectivity means you're now dealing with opinion rather than data."

It is important to recall that, as mentioned previously, user conceptual models may not be appropriate models for use from the point of view of the system designer or information architect, perhaps representing a simplified view of the actual data contained within the system. The actual model in use may be a more complex model that may be visually simplified or 'folded' into an apparently simpler form to approximate the users' expectations, yet itself fulfils the stated aims of flexibility, adequacy to system requirements, and long-term viability.

3.3. Resources, Costs and Benefits

The physical resources required for multilayered prototyping are not as immediately inexpensive as a pure card sorting approach, which can be completed at almost no cost. Sticky notes and non-permanent overhead projector pens are commonplace in the office environment. Large acetate sheets, by comparison, must be procured from specialist shops such as art suppliers. That said, the overall cost of resources for this method remains low, and the startup cost is incurred only once. Materials can be cleaned and reused indefinitely.

There is a widespread perception that usability studies are time-consuming and expensive in general terms. Whilst no detailed cost-benefit analysis will be attempted here, it is interesting to consider the case of the application profile development, engineering and deployment process. Engineering interfaces and internal data models that suit the application profile as published will incur a significant cost for implementers, of which there may be several; deployment of the resulting systems may risk alienating users, a reduction in use and decreased traffic for a service; resources may be swept into improving the interface. Any alterations that are recommended for the application profile itself may result in expensive development and maintenance for each organisation creating or making use of the data.

The benefit arising from user evaluation that is generally cited in business cases primarily reflects the avoidance of costs that might otherwise be incurred at a later date. It may also reflect the possibility of encouraging a larger audience of users through recommendation, satisfaction, repeat visits and word-of-mouth. Rajanan and Iivari (2007), however, note that this benefit is not always well-understood, and that usability may instead be seen as an increased cost. As the role of usability testing in the world of application profile development is not yet clearly understood or widely researched, it is difficult to speculate on whether the approach will improve things materially.

Our present results show that the specific approach described here does provide users with a means by which to develop and their understanding of a domain, and by that means, through examples and provided or negotiated scenarios, to express and refine a conceptual model. This alone is sufficient cause to explore the use of the method further and to explore its use for various purposes, such as forming a basis for eliciting and prioritising simplified models for interface design, or eliciting and prioritising functional requirements for interface design (Snyder, 2003).

3.4. Clarifying Requirements: What's In A Model?

One clear limitation of paper prototyping is worth highlighting; as mentioned previously, unless the process is managed carefully, there is a possibility that the models produced will be over-enthusiastically engineered, containing extraneous detail and information that is of relevance only to implementers. This is particularly the case with domain experts and those with extensive experience of one form or another of data modeling. To some extent, this is unavoidable; however, the phenomenon is an interesting one, as it returns us to the earlier discussion of the EDM concepts of conceptual and logical data models, and causes us to ask: what's in an application profile?

A great deal of additional data is collected during an exploratory or evaluative process, from preferred terminologies, groupings, browse methods and concept models to candidate scenarios. Much of this information does not at present have any obvious role in documenting a Dublin Core Application Profile, leading us to remark that the DCAP itself as an engineering artifact

represents a small (if key) subsection of the information required to develop relevant infrastructure and interactive elements in order to bring a complex application profile into practical use.

Where application profiles contain only a set of terms connected to a single entity, the potential for misunderstanding and confusion is vastly lower than the gap between information architect and developer in the case of more complex constructions, and this brevity represented a form of functional simplicity. Today, there is a possibility that this factor may negatively impact the uptake of otherwise valuable application profiles. One suggestion for future work in this area, therefore, is to examine through a test case the process of encouraging adoption of novel application profiles in some detail, and to identify and provide relevant documentation. Adoption of a technical artifact is a social process, and as a result, communication is a key factor in its success. One important factor in this is the decision to consider social factors such as documentation, interface, audience and usability concerns as integrated parts of a larger general process; evaluation after the event is often too little, too late.

4. Conclusion and Further Work

Early studies in application of this approach suggests that participants are willing to engage with each stage of the process and and that the resulting artifact can represent a useful 'bridge' for communication between participant and investigator. We intend to make use of this approach along with associated work in fast interface prototyping and user testing on the basis of application profile definitions, as part of a wider project in the area of application profile evaluation and engineering recommendations, specifically identification and examination of obstacles delaying uptake of the Scholarly Works Application Profile. We also intend to further explore the potential uses of this approach as a hands-on aid to teaching and learning about metadata. As part of this work, we expect to assess the methods described here via a number of metrics, in particular an estimate of minimum participant numbers, required in order to allow the necessary timescale and overall cost of evaluation via this methodology to be established.

Acknowledgements

This work was partly supported by the JISC

References

American National Standards Institute (1975). ANSI/X3/SPARC Study Group on Data Base Management Systems; Interim Report. FDT (Bulletin of ACM SIGMOD) 7:2.

Beyer, Hugh and Karen Holtzblatt. (1998). Contextual Design: Defining Customer-Centered Systems. San Francisco: Morgan Kaufmann. ISBN: 1-55860-411-1

Hampton, James.A., 1988. Disjunction in natural categories. Memory and Cognition 16, pp. 579–591

Kawakita, Jiro. (1991). The Original KJ Method, Kawakita Research Institute.

Khatri, Vijay, Vessey, Iris, Ramesh, V., Clay, Paul, and Sung-Jin Park (2006). Understanding Conceptual Schemas: Exploring the Role of Application and IS Domain Knowledge. INFORMATION SYSTEMS RESEARCH. Vol. 17, No. 1, March 2006, pp. 81-99. DOI: 10.1287/isre.1060.0081

Rajanen, Mikko and Netta Iivari (2007). Usability Cost-Benefit Analysis: How Usability Became a Curse Word?. In: Baranauskas, Cecilia, Palanque, Philippe, Abascal, Julio and Barbosa, Simone Diniz Junqueira (eds.) Proceedings of the INTERACT 2007 2007, Rio de Janeiro, Brasil. pp. 511-524.

Nielsen, Jakob., and Rolf Molich. (1990). Heuristic evaluation of user interfaces, Proc. ACM CHI'90 Conf. (Seattle, WA, 1-5 April), 249-256.

Rosch, Eleanor, Mervis, Carolyn B. Gray, Wayne D., Johnson, David M. and Penny Boyes-Braem. (1976). Basic objects in natural categories. Cognitive Psychology 7, pp. 573–605.

Rose, Anne, Plaisant, Catherine, and Ben Shneiderman. (1995). Using Ethnographic Methods In User Interface Re-engineering. Proceedings of the ACM 1995 conference for Designing Interactive Systems: processes, practices, methods, and techniques. New York, NY: ACM Press, 1995, pp. 115-122. http://delivery.acm.org/10.1145/230000/225447/p115-rose.pdf

Sinha, Rashmi (2003). Beyond cardsorting: Free-listing methods to explore user categorizations. Retrieved April 2nd from http://www.boxesandarrows.com/view/beyond_cardsorting_free_listing_methods_to_explore_user_categorizations

Sinha, Rashmi R. and Jonathan Boutelle (2004). Rapid information architecture prototyping. Conference on Designing Interactive Systems 2004: 349-352

Snyder, Caroline (2003). Paper Prototyping. Morgan Kaufmann. ISBN 1558608702

Spencer, Donna. (2003). Card-based Classification Evaluation. Retrieved April 2nd from http://www.boxesandarrows.com/view/card_based_classification_evaluation

Spencer, Donna and Todd Warfel (2007). Card Sorting: A Definitive Guide. Retrieved April 2nd from http://www.boxesandarrows.com/view/card_sorting_a_definitive_guide

Sousa, Kenia. Furtado, Elizabeth and Hildeberto Mendonça (2005). UPi – A Software Development Process Aiming at Usability, Productivity and Integration. CLIHC'05, October 23-26, 2005, Cuernavaca, México.

Sutcliffe, Alistair. (2003). Symbiosis and synergy? scenarios, task analysis and reuse of HCI knowledge. Interacting with Computers. Volume 15, Issue 2, April 2003, Pages 245-263